AD 404 715

UNCLASSIFIED

FOR ERRATA

AD 404715

THE FOLLOWING PAGES ARE CHANGES

TO BASIC DOCUMENT

AD-404715

404715

REVISION

Boeing Scientific Research Laboratories Document D1-82-0204, November

AN IBM 7090 PROGRAM FOR COMPUTING TWO-DIMENSIONAL AND AXIALLY SYMMETRIC ISOENERGETIC SUPERSONIC FLOW

by

F. E. Ehlers

* * *

Page 34

Instruction 17 of SUBROUTINE AXIS should be replaced by the following two instructions:

```
17 HP = HPLMIF(BETA(2),ZETA(2),1)
   H2 = HP-ZETA(2)
```

The fourth instruction following instruction 25 should be replaced by:

```
BNEW = (E2*ZETA(2)-G2*(S(1)-S(2))-H2/2.)/F2+BETA(2)
```

Page 28

Instruction 130 should be corrected to read:

```
130 IF(ABSF((Z3-ZNEW)/ZNEW)-EPS)140,140,200
```

June 1965

63-3-5

D1-82-0204

"Also available from the author"

BOEING SCIENTIFIC RESEARCH LABORATORIES

An IBM 7090 Program for Computing Two-Dimensional and Axially-Symmetric Supersonic Flow of an Ideal Gas

F. Edward Ehlers

Mathematics Research

November 1962

FOR ERRATA

AD 404715

THE FOLLOWING PAGES ARE CHANGES

TO BASIC DOCUMENT

REVISION

Boeing Scientific Research Laboratories Document D1-82-0234, November 1962

An IBM 7090 Program for Computing Two-Dimensional and Axially-Symmetric Supersonic Flow of an Ideal Gas

by
F. Edward Ehlers

****

On continuation card 2 of instruction 40 of table 3, page 19,

S(J + 1) should be replaced by S.

The following revised SUBROUTINE READ to replace page [illegible] avoids the computation of $\sqrt{(\mathrm{FREEM})^2 - 1}$ when the upper boundary is not a shock (i.e. IU $\neq$ 4). (FREEM is the Mach number before the shock.)

This program was recoded by Neil Campbell, Airplane Division, The Boeing Company.

****

```
      SUBROUTINE READ
      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N,CODE,IL,IU,A,EPS,GAMMA,X,Y,BETA,ZETA,S,BETAO,XI,STOP
     1,MXSTP
      BMACHF(A) = SQRTF(A*A-1.)
    1 FORMAT(6I3,4E13.0)
    2 FORMAT(4E15.8/E15.8)
    3 FORMAT(E10.0)
    4 FORMAT(43H1ERROR STOP. INITIAL FLOW IS SUBSONIC. M = F12.6)
      READ INPUT TAPE 5,1,M,ICODE,IU,IL,MAX,MXSTP,EPS,GAMMA,FREEM,ANGLE
      READ INPUT TAPE 5,3,STOP
      EPS = EPS/100.
      N=M-1
      CODE = ICODE
      A = MAX
      IF (GAMMA)10,10,15
   10 GAMMA = 1.4
   15 IF (IU-4)31,20,31
   20 IF (FREEM-1.) 1000, 30, 30
 1000 WRITE OUTPUT TAPE 6, 4, FREEM
   99 CALL EXIT
      GO TO 99
   30 BETAO = BMACHF(FREEM)
      XI = TANF(ANGLE)
      GO TO 32
   31 BETAO=0.
      XI=0.
   32 DO 100 I=1,N
  100 READ INPUT TAPE 5,2,X(I),Y(I),BETA(I),ZETA(I),S(I)
      N=M+1
      READ INPUT TAPE 5,2,X(N),Y(N),BETA(N),ZETA(N),S(N)
      RETURN
      END
```

D1-82-0204

AN IBM 7090 PROGRAM FOR COMPUTING TWO-DIMENSIONAL AND AXIALLY-SYMMETRIC SUPERSONIC FLOW OF AN IDEAL GAS

by

F. Edward Ehlers

Mathematical Note No. 272

Mathematics Research Laboratory

BOEING SCIENTIFIC RESEARCH LABORATORIES

November 1962

The author wishes to acknowledge the valuable assistance of Thomas A. Bray in the coding and checking of the subroutines.

TABLE of CONTENTS

REFERENCES

[1] Ehlers, F. Edward. The Method of Characteristics for Isoenergetic Supersonic Flows Adapted to High-Speed Digital Computers. J. Soc. Indust. Appl. Math. Vol 7, p.85, March,1959.

[2] Ehlers, F. Edward. The Method of Characteristics Applied to the Design of Supersonic Axially-Symmetric Nozzles. Boeing Document D2-2118, August 2, 1957.

[3] Handbook of Supersonic Aerodynamics. Navord Report 1488 (vol. 2) October 1, 1950.

[4] Gray, Malcolm D. Wind Tunnel Nozzle (NOZZLE) Boeing Document AS 1402, January 1963.

[5] Ehlers, F. Edward. IBM 701 Program JA01 for Computing Two-Dimensional Supersonic Nozzles. Boeing Document D2-1411, December 27, 1956.

1. Introduction

In an article published in the Journal of the Society for Industrial and Applied Mathematics (Ref [1]), the writer formulated the method of characteristics for the supersonic isoenergetic flow of an ideal gas specifically for coding on a high speed digital computer. By the introduction of variables $\beta = \sqrt{M^2 - 1}$ and $\zeta = \tan \Theta$, where M is the local Mach number and Θ is the local flow angle, the coefficients of the characteristic equations were simplified to algebraic rational functions, thereby appreciably reducing the required computing time. This document describes the program coded for the IBM 7090 electronic computer for the method of Ref [1]. By means of this program, it is possible to compute the axially-symmetric or two dimensional Mach net for supersonic flows over pointed bodies with attached shocks, through channels consisting of two rigid walls, and through nozzles with an axis of symmetry. The computation of flows with free streamline boundaries is also possible, if no shocks occur within the flow.

2. Description of the general procedure for computing the Mach net.

The complete program uses any initial data line which does not coincide with a Mach line to compute the location and flow parameters for the next set of points downstream of this initial line. To start the computations, the flow data along this initial line must be provided and must include the data for two boundary points at the ends of the line.

The scheme of storing the points is illustrated in the drawing of Fig. 1. Note that the lower boundary point data is denoted by the subscript 1. Interior points of the Mach net are computed by a subroutine denoted by ITER (with ITER 1). By means of the CHAR (k) subroutine, ITER for odd numbered data lines takes the top boundary point denoted by N + 1 and the next lower point, N - 1, computes the next downstream point, and stores it in the location for the point N. For other points on the initial line, ITER takes the consecutive points j and j + 1 , computes a third point, and stores the data in the location for the point j + 1. This procedure is continued down the line of data. Note that the initial line data is cancelled out except for the boundary points and therefore each set of computations must be stored on the output tape for later off line printing. Since the upper boundary point may be needed in the calculation of the next upper boundary point, the initial line is loaded to skip the locations for the point N. Thus, the second line of data (and all even numbered lines thereafter) contains data for N - 1 points located for points 2 consecutively to N. All odd numbered data lines start with the point 1, run consecutively to N - 1, but skip N with the upper boundary point in the location for N + 1.

When even numbered data lines are used to compute the next downstream data lines, the upper and lower boundary points are computed first and are stored in locations for the points N + 1 and 1, respectively. All remaining points are computed by using data points j and j + 1 with the new data stored in the point j locations.

3. Designation of Types of Flow.

The program computes the Mach net for either isentropic or non-isentropic flow and for either axially-symmetric or two dimensional flow according to a specified code. This flow code is an integer from 1 to 4 denoted in the program by ICODE or CODE depending upon the form used by the program. The interpretation of these codes is given in Table I.

TABLE I

ICODE	TYPE OF FLOW
1	Axially symmetric with variable entropy
2	Axially symmetric and isentropic
3	Two dimensional with variable entropy
4	Two dimensional and isentropic

4. Designation of upper and lower boundary points.

The selections of the subroutines for the upper and lower boundaries are made by code numbers denoted by IU and IL, respectively. The types of lower boundaries provided for are the axis of symmetry, fixed wall, and the free streamline bordering still air. For the upper boundary, the fix wall, the free streamline, and the inclined shock with a uniform flow ahead of it may be chosen. The same code numbers are used for both the upper and lower boundary and are given in Table 2.

TABLE 2

IL or IU	BOUNDARY
1	Axis of symmetry
2	Fixed wall
3	Free streamline
4	Shock preceded by a uniform stream.

Note that codes IL = 4 and IU = 1 are inadmissible codes. When these codes are sensed by the program, the calculations are stopped, a statement of the error is written on the output tape, and calculations are terminated.

5. Preparation of input data cards.

The first input data card contains the code numbers, the number of points and other parameters associated with the flow field. The format of this card is expressed in Fortran language by

```
FORMAT (6I3, 4E13.0)
```

The order of the data on the first card is

1. M. The number of points on the initial data line.
2. ICODE. A number from 1 to 4 designating the type of flow and it is determined according to Table 1 in Section 3.
3. IU. Upper boundary code according to Table 2.
4. IL. Lower boundary code according to Table 2.
5. MAX. Maximum number of iterations to be calculated when the iteration does not converge to the accuracy specified.

6. MXSTP. The maximum number of steps (or data lines) to be computed. This must be an even number, if the last data line is to be punched on cards as the initial line for continuing the Mach net at a later run time.

7. EPS. Maximum percent difference between the values of each quantity from consecutive iterations.

8. GAMMA. The value of the ratio of specific heats for the gas. If space is left blank, GAMMA is set equal to 1.4.

9. FREEM. The free stream Mach number ahead of the shock wave. (Not required unless IU = 4).

10. ANGLE The angle of inclination of the shock at the initial line upper boundary point for code IU = 4.

The second data card with the format (1E10.0) contains STOP, the maximum value of X to which the Mach net is to be computed. If X = STOP is not attained in the number of steps specified by MXSTP, the program writes this information on the tape to be printed at the end of the calculations and punches the last set of data on cards for continuing the Mach net farther downstream in a later run.

The cards for the initial line are loaded in the machine consecutively beginning with the lower boundary point data. The data for each point is punched on two cards according to the format (4E15.8/E15.8). The order of the data is

1. X Axial coordinate of the point.
2. Y Vertical coordinate of the point.
3. BETA The local cotangent of Mach angle. $\beta = \sqrt{M^2 - 1}$, where M is the Mach number.

4. ZETA The local tangent of the flow angle.

5. S The local value of the entropy. ($S = 0$ if the flow is isentropic).

When the boundary code numbers IL = 2, or IU = 2, or both are prescribed, then FORTRAN statements must be included for the y coordinates and the slope of the boundary curves as functions of the axial variable x.

The subroutines for the coordinates and slope, WALL(X,K) and WALPR(X,K) are listed on page 36 of Appendix III for the upper wall given by the parabola

$$y = 1 + x^2/5$$

and for the lower wall, given by

$$y = 0.1 + 0.05004166x - 0.01001042x^2$$

The BOUND (M) subroutine supplies the code $K = 0$ for computing the upper wall and $K = 1$ for the lower wall.

APPENDIX I: Interaction of a Mach Line with a Shock Wave.

Since the procedure for computing the interaction of a Mach line with a shock wave described in Ref.[1] does not always converge we have chosen a different method of iteration, which we shall explain in some detail. Let the quantity at the previous shock point be denoted by the subscript 2 and the nearest interior point by 1 (see Fig. 2). The first estimate for the location of the new point on the shock is given by

$$x_3 = (y_2 - y_1 + A_1^+ x_1 - \xi_2 x_2)/(A_1^+ - \xi_2) \tag{1}$$

$$y_3 = y_1 + A_1^+(x_3 - x_1) \tag{2}$$

where ξ_2 is the slope of the shock at the point (2) and A_1^+ is defined in Ref.[1]. The flow variables at the new point x_3, y_3 must satisfy the compatability relation

$$\begin{aligned} E_1(\zeta_3 - \zeta_1) - F_1(\beta_3 - \beta_1) - G_1(s_3 - s_1) \\ + H_1^+(y_3 - y_1)(1/y_1 + 1/y_3)/2 = 0. \end{aligned} \tag{3}$$

As shown in section 9 of Ref [1], ζ_3, β_3, s_3 are functions of ξ_3 and the Mach number ahead of the shock. Thus the left hand side of Eq.(3) may be regarded as a function of ξ_3, say $f(\xi_3)$, then that value of ξ_3 which satisfies Eq.(3) can be determined by Newton's method, i.e., by iterating

$$\xi_3 = \xi_3 - f(\xi_3)/f'(\xi_3) \tag{4}$$

To evaluate $f(\xi_3)$ and $f'(\xi_3)$ we use the formulas of β_3, ζ_3, and s_3 from Ref.[1]. These are written in the following form:

$$\beta_3^2 = g[-1 + C_1(\xi_3^2 + 1)g\xi_3^2(\beta_0^2 + 1)/(C_1\xi_3^2 + 2)(C_3\xi_3^2 - 1)] \qquad (5)$$

$$\zeta_3 = 2(\beta_0^2\xi_3^2 - 1)/\xi_3(C_1\xi_3^2 + C_2) \qquad (6)$$

$$s_3 = \log[\frac{C_3\xi_3^2 - 1}{g(1 + \xi_3^2)}] + \gamma \log[\frac{(C_1\xi_3^2 + 2)}{(\gamma + 1)\xi_3^2(\beta_0^2 + 1)}] \qquad (7)$$

where

$$\begin{aligned} g &= (\gamma + 1)/(\gamma - 1), \\ C_1 &= (\gamma - 1)\beta_0^2 + \gamma + 1, \\ C_2 &= C_1 + 2(\beta_0^2 + 1), \\ C_3 &= C_1/(\gamma - 1) + g\beta_0^2 . \end{aligned} \qquad (8)$$

Differentiating Eqs. (5) and (6) logarithmically and also Eq. (7), we obtain for the derivatives with respect to ξ_3,

$$\beta_3' = (\beta_3^2 + \frac{\gamma + 1}{\gamma - 1})\xi_3[\frac{2}{\xi_3^2(C_1\xi_3^2 + 2)} - \frac{1 + C_3}{(1 + \xi_3^2)(C_3\xi^2 - 1)}]/\beta \qquad (9)$$

$$\zeta_3' = 2(2\beta_0^2 - C_1\xi_3\zeta_3)/(C_1\xi^2 + C_1) - \zeta_3 / \xi_3 \qquad (10)$$

$$s_3' = 2\xi_3[\frac{1 + C_3}{C_3\xi_3^2 - 1)(1 + \xi_3^2)} - \frac{2\gamma}{\xi_3^2(C_1\xi_3^2 + 2)}] \qquad (11)$$

With Eqs. (9), (10), and (11), $f'(\xi_3)$ is given by

$$f'(\xi_3) = E_1\zeta_3' - F_1\beta_3' - G_1 s_3' . \qquad (12)$$

In the process of iterating Eq. (4), estimates of β_3, S_3, and ζ_3 are obtained as well as of ξ_3. The iteration is continued until the value of ξ_3 converges. The sequence of calculations is repeated by replacing the coefficients of Eq.(3) and (12) by their average values between the points 1 and 3 and between the points 2 and 3. For example F_1 is replaced by $(F_1 + F_3)/2$ in Eqs. (3) and (12) and ξ_3 and A_1^+ are replaced by $(\xi_3 + \xi_2)/2$ and $(A_1^+ + A_3^+)/2$ in Eqs.(1) and (2). An alternate procedure is to evaluate A^+, $E(\zeta)$, $F(\beta)$, etc., for the quantities $(\beta_1 + \beta_3)/2$ and $(\zeta_1 + \zeta_3)/2$.

The program as listed on page 37 Appendix III has an additional feature which evens out the Mach net in the region of the shock. When the Mach net is started from an initial line including a point on the shock, the next point on the shock wave will be farther downstream than the other points on that line of data (compare points 3 and 3' of Fig. 2). This can be remedied by computing the next point on the shock from a new interior point found by interpolating linearly between the two points A and B along the Mach line AB in Fig. 3. Lines BC and ED are chosen to have the slope of the Mach line at B given by

$$A_n = \frac{\beta_n \zeta_n + 1}{\beta_n - \zeta_n} \tag{13}$$

The equations of line DE and AB are

$$y - y_f = A_n(x - x_f) \tag{14}$$

$$y - y_{n+1} = D_1(x - x_{n+1}) \tag{15}$$

respectively, where

$$D_1 = (y_{n+1} - y_n)/(x_{n+1} - x_n). \tag{16}$$

Choosing $x_{n+1} - x_f = 2(x_{n+1} - x_n)$, we can determine y_f by the intersection of line DE with the shock line given by

$$y_{n+1} - y_f = \zeta(x_{n+1} - x_f) \tag{17}$$

where ζ is the slope of the shock at x_{n+1}, y_{n+1}. Solving Eqs. (14) and (15) for x and y and substituting the values of x_f and y_f yields

$$\begin{aligned} x &= x_{n+1} + 2(\zeta - A_n)(x_n - x_{n+1})/(D_1 - A_n) \\ y &= y_{n+1} + D_1(x - x_{n+1}). \end{aligned} \tag{18}$$

By the point (x,y), the segment AB is divided into segments whose lengths have the ratio $D_2 / (1 - D_2)$ where

$$D_2 = (x_{n+1} - x)/(x_{n+1} - x_n).$$

Hence, the values of β, ζ, and s at the point E are given by equations of the form

$$\beta = \beta_n D_2 + \beta_{n+1}(1 - D_2).$$

The next point along the shock is computed by using the values at the interpolated point in place of the values at x_n, y_n for the interior point denoted by the subscript 1 in the foregoing discussion.

APPENDIX II: The Design of Perfect Nozzles

Since the program as described in the main part of this document is capable of computing a symmetric isentropic flow with a fixed wall, it is possible to make use of it in the design of both two dimensional and axially symmetric perfect nozzles. However, an additional program is required to compute the transition region which produces a uniform supersonic flow.

A method for designing supersonic axially symmetric nozzles consisting of a circular throat region, of a conical expanding section, and of a final transition to uniform flow (see Fig.4), is described in Ref.[2]. The assumption that the flow field in the conical region is approximated by the supersonic source leads to a set of equations for finding the transition boundary Mach line. For a nozzle with a cone section of half angle Θ_A and length L and with a throat radius R_T, the coordinates of the transition Mach line are given parametrically by

$$
\begin{aligned}
x/h &= (\tau \cos \Theta - \tau_A \cos \Theta_A)/2\tau_E \sin(\Theta_A/2) \\
&\quad + (R_T/h)\sin \Theta_A + (L/h)\cos \Theta \qquad (19) \\
y/h &= (\tau/\tau_E)\sin \Theta/2\sin(\Theta_A/2) \;.
\end{aligned}
$$

The parameter Θ is the local flow angle, h is the throat radius, and τ is related to the local value of β by

$$\tau = (1 + \beta^2/k)^{k/4}/(\beta^2 + 1)^{1/4} \qquad (20)$$

where

$$k = (\gamma + 1)/(\gamma - 1).$$

The subscripts A and E denote quantities evaluated at the points A and E of Fig. 4, respectively. The quantities β and $\zeta = \tan\theta$ along the Mach line are functionally related by

$$\theta = (\omega_E - \omega)/2 \tag{21}$$

where ω is the Prandtl-Meyer angle,

$$\omega = \sqrt{k}\,\tan^{-1}(\beta/\sqrt{k}) - \tan^{-1}\beta \tag{22}$$

which is tabulated in Ref.[3] page 504 and following. With Eqs. (19) through (22), the values of x, y, β, and ζ for points on the transition boundary Mach line are determined parametrically by a choice of β. Dividing the range of β_A to β_E into N, say, intervals, we obtain a smoothly spaced set of points along the boundary Mach line.

By means of the foregoing procedure, or by means of the program described in the main section of the document, we can establish values of β, ζ, x, and y at $N + 1$ regularly spaced points along the Mach line AE in Fig. (4). Since, we specify uniform flow at the exit end of the nozzle, the values $\beta = \beta_E$ and $\zeta = 0$ are known along Mach line EC and x and y are then related by

$$\beta_E y = x - x_E .$$

With conditions specified along the two Mach lines AE and EC, the flow is uniquely determined in a region AECB bounded by the four Mach lines shown in Fig. 4.

By working upward step by step along the Mach line AE from a point on the line EC, we calculate the next Mach line of the same family (see Fig.5). This is easily accomplished by using, for example, the simple program suggested in Table 3, page 19, together with the subroutine ITER and with the STREM routine programmed by Malcolm Gray [4] and listed in Table 4, page 20.

The subroutine STREM utilizes a method of polynomial interpolation to find the streamline point on the next downstream Mach line. Three points are chosen so that the y coordinate of the highest point is greater than the value of y at the last streamline point, x_0, y_0. Denoting the points on the Mach line by subscripts 1,2, and 3, (see Fig.6) and applying Lagranges' interpolation formula yield the following equations for $y(x)$, $\zeta(x)$, and $\beta(x)$ along the Mach line;

$$y(x) = y_1L_1(x) + y_2L_2(x) + y_3L_3(x)$$

$$\zeta(x) = \zeta_1L_1(x) + \zeta_2L_2(x) + \zeta_3L_3(x)$$

$$\beta(x) = \beta_1L_1(x) + \beta_2L_2(x) + \beta_3L_3(x)$$

where

$$L_1(x) = (x - x_2)(x - x_3)/(x_1 - x_2)(x_1 - x_3)$$

$$L_2(x) = (x - x_1)(x - x_3)/(x_2 - x_1)(x_2 - x_3)$$

$$L_3(x) = (x - x_1)(x - x_2)/(x_3 - x_1)(x_3 - x_2)$$

The line with the slope $(\zeta_0 + \zeta(x))/2$, representing the streamline sgment is given by

$$F(x) = y - y_0 - [\zeta(x) + \zeta_0](x - x_0)/2 = 0$$

The solution by Newton's method of this last equation with $y = y(x)$ along the Mach line yields the next downstream point on the boundary streamline of the transition region.

The program is capable of using up to 50 points along the Mach line. The variables X(I), Y(I), Z(I), B(I) denote the Cartesian coordinates of the points on the Mach line and their corresponding values of ζ' and β, respectively. The quantities XYBZO(I), I = 1,2,3, and 4, denote the values of X, Y, β, and ζ, respectively, for the last streamline point, and XYBZ1(I), the corresponding values for the point to be computed on the streamline. The symbols XY(1) and XY(3) represent the coordinates x,y of the point E in Fig. 4, and XY(2) and XY(4), for the point C.

The code number IDONE = 1 when the program is entered. At exit, the code number may be changed and then has the following interpretations:

IDONE	INTERPRETATION
1	Last streamline y coordinate is less than the largest value of y on the Mach line. Next point of the boundary is successfully computed.
2	Last streamline y coordinate is greater than the largest value of y on the Mach line. New point is not determined.

IDONE	INTERPRETATION
3	Smallest y on Mach line is greater than y of previous streamline point. This indicates that the next streamline point is on Mach line EC of Fig. 4 and the point is determined under this assumption.
4	The iteration for x did not converge to the required relative accuracy EPSIL in 10 iterations.

The number N is set equal to the number of the lowest point for which the y coordinate is greater than the previously computed boundary point. The transition region program in Table 3 uses this feature of STREM to reduce systematically the number of points computed for each successive Mach line.

To compute two dimensional nozzles, relations similar to Eqs. (19) through (22) can be derived for the boundary Mach line, or with a known boundary for the throat region of the nozzle the Mach net can be calculated by the program described in Sections 1 to 5. By assuming that the lines of constant Mach number are circles and are derived from a source flow, one can obtain the initial line data (see Ref.[3], and Fig. 7). The Mach number is found from the ratio of the area of the initial line surface to the throat area. Calculation of the transition region for a two dimensional nozzle is much simpler since the region consists of one family of straight Mach lines along which the flow properties are constant. By approximating the streamline boundary between two Mach lines (see Fig. 8) by a straight line whose slope is the average of the slopes on the two Mach lines, we can determine the location of next boundary point by solving two simultaneous linear equations (see Ref.[5]). In this way, we obtain

$$x = [y_{n+1} - y_0 + (\zeta_n + \zeta_{n+1})x_0/2 - A_{n+1}x_{n+1}]/[(\zeta_n + \zeta_{n+1})/2 - A_{n+1}]$$

$$y = y_{n+1} + A_{n+1}(x - x_{n+1})$$

where

$$A_{n+1} = (\beta_{n+1}\zeta_{n+1} + 1)/(\beta_{n+1} - \zeta_{n+1})$$

and the subscripts denote the values of the quantities at the corresponding points shown in Fig. 8. A program suggested for computing the Prandtl-Meyer region is given in Table 5. Note that the data for the boundary Mach line are read beginning at the upper and upstream end of the line. The point 1 must also be a boundary point on the fixed wall.

TABLE 3

```
C     TRANSITION REGION      AXIALLY SYMMETRIC NOZZLE
      DIMENSION X(50),Y(50),BETA(50),ZETA(50),XYBZ0(4),XYBZ1(4),XY(4)
   10 READ INPUT TAPE 5,20,  M,N,EPS,DX,GA
   20 FORMAT (2I3,3E16.8)
      READ INPUT TAPE 5,30,(XYBZ0(I),I=1,4),(XY(I),I=1,4),(X(I),Y(I),BET
     1A(I),ZETA(I),I=1,N)
   30 FORMAT (4E16.8)
      KODE = 2.
      IDONE=1
      S=0.
      XE=X(1)
      NN=N-1
      DO 60 I=1,M
      X(1)=X(1)+DX
      Y(1)=(X(1)-XE)/BETA(1)
      DO 40 J=1,NN
   40 CALL ITER(BETA(J+1),BETA(J),ZETA(J+1),ZETA(J),S,S,X(J+1),X(J),
     1 Y(J+1),Y(J),KODE,N,EPS,GA,BETA(J+1),ZETA(J+1),S(J+1),X(J+1),Y(J+1
     1))
      N=NN+1
      CALL STREM(N,IDONE,EPS,X,Y,BETA,ZETA,XYBZ0,XYBZ1,XY)
      NN=N-1
      WRITE OUTPUT TAPE 6,30,(XYBZ1(I),I=1,4)
      DO 50 K=1,4
   50 XYBZ0(K)=XYBZ1(K)
      GO TO (60,70,10,90),IDONE
   60 CONTINUE
      GO TO 10
   70 WRITE OUTPUT TAPE 6,80
   80 FORMAT(42H ERROR RETURN FROM STREM--K GREATER THAN N)
      GO TO 10
   90 WRITE OUTPUT TAPE 6,100
  100 FORMAT(52H ERROR RETURN FROM STREM--ITERATION DID NOT CONVERGE)
      GO TO 10
      END
```

INPUT DATA

M = Number of intervals along straight Mach line EC in Fig. 4.

N = Number of points along transition boundary Mach line AE in Fig. 4.

DX = Interval length in x for points along EC.

GA = γ, the ratio of specific heats for the gas.

TABLE 4

```
      SUBROUTINE STREM (N, IDONE, EPSIL, X, Y, B, Z, XYBZ0, XYBZ1, XY)
      DIMENSION X(50), Y(50), B(50), Z(50), XYBZ0(4), XYBZ1(4), XY(4)
      INDIC = 1
      K = 1
    1 IF (XYBZ0(2) - Y(K)) 2, 4, 8
    2 IF (K - 1) 18, 18, 3
    3 K = K + 1
      GO TO 20
    4 IF (XYBZ0(4)) 5, 5, 7
    5 XYBZ1(1) = X(K)
      XYBZ1(2) = Y(K)
      XYBZ1(3) = B(K)
      XYBZ1(4) = 0.
      IF (K - 1) 6, 6, 21
    7 IF (K - 2) 18, 18, 8
    8 K = K + 1
      IF (K - N) 1, 9, 19
    9 IF (XYBZ0(4) - Z(N)) 20, 20, 19
   19 IDONE = 2
      GO TO 17
   20 N = K
      X1MX2 = X(K) - X(K-1)
      X1MX3 = X(K) - X(K-2)
      X2MX3 = X(K-1) - X(K-2)
      G1DEN = X1MX2*X1MX3
      G2DEN = -X1MX2*X2MX3
      G3DEN = X1MX3*X2MX3
      H =(Y(K) - Y(K-1))/X1MX2
      X1OLD = (XYBZ0(2) - Y(K) + H*X(K) - XYBZ0(1)*XYBZ0(4))/(H - XYBZ0(
     14))
      LOOP = 1
   10 TERM1 = X1OLD - X(K)
      TERM2 = X1OLD - X(K-1)
      TERM3 = X1OLD - X(K-2)
      G1 = (TERM2*TERM3)/G1DEN
      G2 = (TERM1*TERM3)/G2DEN
      G3 = (TERM1*TERM2)/G3DEN
      XYBZ1(2) = Y(K)*G1 + Y(K-1)*G2 + Y(K-2)*G3
      XYBZ1(4) = Z(K)*G1 + Z(K-1)*G2 + Z(K-2)*G3
      GO TO (11, 16), INDIC
   11 G1PRI = (TERM2 + TERM3)/G1DEN
      G2PRI = (TERM1 + TERM3)/G2DEN
      G3PRI = (TERM1 + TERM2)/G3DEN
      YPRIM = Y(K)*G1PRI + Y(K-1)*G2PRI + Y(K-2)*G3PRI
      H = X1OLD - XYBZ0(1)
      ZPRIM = (Z(K)*G1PRI + Z(K-1)*G2PRI + Z(K-2)*G3PRI)*H/2.0
      ZFRAC = (XYBZ0(4) + XYBZ1(4))/2.0
      X1NEW = X1OLD - (XYBZ1(2) - XYBZ0(2) - H*ZFRAC)/(YPRIM - ZPRIM -
     1ZFRAC)
      IF ((ABSF(X1NEW - X1OLD)/X1NEW) - EPSIL) 12, 12, 13
```

```
   12 INDIC = 2
      XYBZ1(1) = X1NEW
      GO TO 15
   13 LOOP = LOOP + 1
      IF (LOOP - 10) 15, 15, 14
   14 IDONE = 4
      GO TO 17
   15 X1OLD = X1NEW
      GO TO 10
   16 XYBZ1(3) = B(K)*G1 + B(K-1)*G2 + B(K-2)*G3
   17 RETURN
   18 SLOPE = (XY(4) - XY(3))/(XY(2) - XY(1))
      XY5 = XYBZ0(4)/2.0
      XYBZ1(1) = (XY(3) - SLOPE*XY(1) - XYBZ0(2) + XY5*XYBZ0(1))
     1/(XY5 - SLOPE)
      XYBZ1(2) = XY5*(XYBZ1(1) - XYBZ0(1)) + XYBZ0(2)
      XYBZ1(3) = B(1)
      XYBZ1(4) = 0.0
    6 IDONE = 3
      GO TO 17
   21 N = K
      GO TO 17
      END
```

TABLE 5

```
C     PRANDTL-MEYER REGION
      DIMENSION X(50),Y(50),BETA(50),ZETA(50),XB(50),YB(50)
      APLMIF(A,C,M) = (A*C+(-1.)**M)/(A-(-1.)**M*C)
   10 READ INPUT TAPE 5,20,N
   20 FORMAT ( I3 )
      READ INPUT TAPE 5,30,(X(I),Y(I),BETA(I),ZETA(I), I=1,N)
   30 FORMAT (4F12.8)
      XB(1)= X(1)
      YB(1)=Y(1)
      DO 40 I=2,N
      A1 = APLMIF(BETA(I),ZETA(I),0)
      Z1 = (ZETA(I) + ZETA(I-1))/2.
      XB(I) = (YB(I) - YB(I-1) + Z1*XB(I) - A1*X(I))/(Z1-A1)
   40 YB(I) = Y(I) + A1*(XB(I) - X(I))
      WRITE OUTPUT TAPE 6,30,(XB(I),YB(I),BETA(I),ZETA(I), I=1,N)
      GO TO 10
      END
```

INPUT DATA

N = Number of points along boundary Mach line of Prandtl-Meyer Region.

APPENDIX III <u>FORTRAN Listing of Subroutines for the Method of Characteristics</u>

```
C       MAIN PROGRAM

        DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
        COMMON N,CODE,IL,IU,A,EPS,GAMMA,X,Y,BETA,ZETA,S,BETA0,XI,STOP
       1,MXSTP
   80   CALL READ
        N=N
        MXSTP=MXSTP
        CALL WRITE1
        CALL WRITE2(1)
        DO  100  I=1, MXSTP
        IF ((-1)**I) 10, 10, 20
   10   K = 1
        GO TO 30
   20   K = 0
   30   CALL CHAR(K)
        CALL WRITE 2(I+1)
        DO 60 J=1, N
        IF (X(J)-STOP) 100, 60, 60
   60   CONTINUE
        GO TO 80
  100   CONTINUE
   70   WRITE OUTPUT TAPE 6, 1
    1   FORMAT(64H THE LOWER LIMIT WAS NOT REACHED IN THE MAXIMUM NUMBER O
       1F STEPS.)
        M=N-2
        DO 150 I=1,M
  150   WRITE OUTPUT TAPE 14,200,X(I),Y(I),BETA(I),ZETA(I),S(I)
  200   FORMAT(4E15.8/E15.8)
        WRITE OUTPUT TAPE14,200,X(N),Y(N),BETA(N),ZETA(N),S(N)
        GO TO 80
        END
```

```
      SUBROUTINE READ

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N,CODE,IL,IU,A,EPS,GAMMA,X,Y,BETA,ZETA,S,BETAO,XI,STOP
     1,MXSTP
      BMACHF(A) = SQRTF(A*A-1.)
    1 FORMAT(6I3,4E13.0)
    3 FORMAT(E10.0)
      READ INPUT TAPE 5,1,M,ICODE,IU,IL,MAX,MXSTP,EPS,GAMMA,FREEM,ANG
      READ INPUT TAPE 5,3,STOP
      EPS = EPS/100.
      N=M-1
      CODE = ICODE
      A = MAX
      IF (GAMMA)10,10,15
   10 GAMMA = 1.4
   15 IF (IU-4)30,20,30
   20 IF (FREEM-1.) 1000, 30, 30
   30 BETAO = BMACHF(FREEM)
      XI = TANF(ANGLE)
      DO 100 I=1,N
  100 READ INPUT TAPE 5,2,X(I),Y(I),BETA(I),ZETA(I),S(I)
    2 FORMAT(4E15.8/E15.8)
      N=M+1
      READ INPUT TAPE 5,2,X(N),Y(N),BETA(N),ZETA(N),S(N)
      RETURN
 1000 WRITE OUTPUT TAPE 6, 4, FREEM
      CALL EXIT
    4 FORMAT(43H1ERROR STOP. INITIAL FLOW IS SUBSONIC. M = F12.6)
      END
```

```
      SUBROUTINE WRITE 1
      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N,CODE,IL,IU,A,EPS,GAMMA,X,Y,BETA,ZETA,S,BO,XI,STOP,
     1MXSTP
      J = CODE
      WRITE OUTPUT TAPE 6, 1
      GO TO (10,20,30,40), IU
   10 WRITE OUTPUT TAPE 6, 2
      GO TO 50
   20 WRITE OUTPUT TAPE 6, 3
      GO TO 50
   30 WRITE OUTPUT TAPE 6, 4
      GO TO 50
   40 BMACH = SQRTF(BO*BO+1.)
      WRITE OUTPUT TAPE 6, 5, BMACH
   50 GO TO (60,70,80), IL
   60 WRITE OUTPUT TAPE 6, 6
      GO TO 90
   70 WRITE OUTPUT TAPE 6, 7
      GO TO 90
   80 WRITE OUTPUT TAPE 6, 8
   90 GO TO (100,100,110,110), J
  100 WRITE OUTPUT TAPE 6, 9
      GO TO 120
  110 WRITE OUTPUT TAPE 6, 11
  120 GO TO (130,140,130,140), J
  130 WRITE OUTPUT TAPE 6, 12
      GO TO 150
  140 WRITE OUTPUT TAPE 6, 13
  150 PCTER = EPS*100.
      WRITE OUTPUT TAPE 6,14,PCTER,MXSTP,STOP
      RETURN
    1 FORMAT(1H1///39X,42HCHARACTERISTIC SOLUTION OF SUPERSONIC FLOW)
    2 FORMAT(///35X,27HILLEGAL UPPER BOUNDARY CODE)
    3 FORMAT(///35X,27HUPPER BOUNDARY   FIXED WALL)
    4 FORMAT(///35X,28HUPPER BOUNDARY   FREE STREAM)
    5 FORMAT(///35X,42HUPPER BOUNDARY   SHOCK   INITIAL MACH NO. ,E15.8)
    6 FORMAT(//35X,33HLOWER BOUNDARY   AXIS OF SYMMETRY)
    7 FORMAT(//35X,27HLOWER BOUNDARY   FIXED WALL)
    8 FORMAT(//35X,28HLOWER BOUNDARY   FREE STREAM)
    9 FORMAT(//35X,22HAXIALLY SYMMETRIC FLOW)
   11 FORMAT(//35X,20HTWO DIMENSIONAL FLOW)
   12 FORMAT(//35X,12HISOENERGETIC)
   13 FORMAT(//35X,10HISENTROPIC)
   14 FORMAT(//35X,12HNO MORE THAN,F11.6,33H PERCENT ERROR WILL BE INTRO
     1DUCED,//35X,12HIN ONE STEP.///35X,34HTHE PROCESS WILL BE STOPPED A
     2FTER ,I4,19H STEPS IF THE LOWER,//35X,11HLIMIT, X = F12.6,17H, IS
     3NOT REACHED.)
      END
```

```
      SUBROUTINE WRITE2(K)

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, IL, IU, A, EPS, GAMMA, X, Y, BETA, ZETA, S, BO, XI
      I = 1.5+.5*(-1.)**K
   10 WRITE OUTPUT TAPE 6, 1, K
      M=N-2
      DO 1000 J=1,M
      II=I+J-1
  750 L=J
 1000 WRITE OUTPUT TAPE 6,2,L,X(II),Y(II),BETA(II),ZETA(II),S(II)
      IF ((-1)**K) 2000,2000,2050
 2000 L=N-1
      WRITE OUTPUT TAPE 6,2,L,X(N),Y(N),BETA(N),ZETA(N),S(N)
      IF (IU-4) 2050,2020,2050
 2020 ANGLE = ATANF(XI)
      WRITE OUTPUT TAPE 6, 4, ANGLE
 2050 RETURN
    1 FORMAT(1H1/39X,42HCHARACTERISTIC SOLUTION OF SUPERSONIC FLOW,///54
     1X,8HSTEP NO.,I4/////,16X,1HX,22X,1HY,18X,9HBETA     ,13X,11HZETA
     2    ,14X,7HENTROPY,/120X)
    2 FORMAT(120X/I5,5E23.8)
    4 FORMAT(120X/41H THE SHOCK ANGLE AT THE BOUNDARY POINT IS,E20.8)
      END
```

```
      SUBROUTINE CHAR(K)

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, LOW, HIGH, A, EPS, GAMMA, X, Y, BETA, ZETA, S
      M = N-2
      IF (K) 1000, 10, 100
   10 CALL BOUND(1)
      CALL BOUND(2)
      IF DIVIDE CHECK 40, 50
   40 WRITE OUTPUT TAPE 6, 2
    2 FORMAT(//23H DIVIDE CHECK IN BOUND )
   50 CONTINUE
      DO 15 J=2, M
   15 CALL ITER1(J,(J+1),J)
      IF DIVIDE CHECK 20, 30
   20 WRITE OUTPUT TAPE 6, 3
    3 FORMAT(23H DIVIDE CHECK IN ITER  )
   30 CONTINUE
      RETURN
  100 CALL ITER1(N-2,N,N-1)
      IF DIVIDE CHECK 110, 120
  110 WRITE OUTPUT TAPE 6, 3
  120 CONTINUE
      DO 150 J=2, M
      I = M+2-J
  150 CALL ITER1(I-1,I,I)
      IF DIVIDE CHECK 160,170
  160 WRITE OUTPUT TAPE 6, 3
  170 CONTINUE
      RETURN
 1000 CALL EXIT
      END
```

```
      SUBROUTINE ITER1 (I,J,K)

      DIMENSION X(1000),BETA(1000),ZETA(1000),S(1000),Y(1000)
      COMMON N,CODE,IL,IU,A,EPS,GAMMA,X,Y,BETA,ZETA,S
      MAX=A
      KODE=CODE
      CALL ITER(BETA(I),BETA(J),ZETA(I),ZETA(J),S(I),S(J),X(I),X(J),Y(I)
     1,Y(J),KODE,MAX,EPS,GAMMA,BETA(K),ZETA(K),S(K),X(K),Y(K))
      IF(SENSE LIGHT 4)10,30
   10 WRITE OUTPUT TAPE 6,20,I,J,K
   20 FORMAT(30H ITER DOES NOT CONVERGE FOR I=I5,4H  J=I5,4H  K=I5)
   30 CONTINUE
      RETURN
      END
```

INTERIOR POINT ITERATION ROUTINE

```
      SUBROUTINE ITER(B1,B2,Z1,Z2,S1,S2,X1,X2,Y1,Y2,ICODE,MAX,EPS,G,BN3,
     1ZN3,SN3,XN3,YN3)

      FUNCTIONS

      EFNF(Z) = 1.0/(1.0+Z**2)
      APLF(B,Z) = (B*Z+1.0)/(B-Z)
      AMIF(B,Z) = (B*Z-1.0)/(B+Z)
      FFNF(B)=(2.0*B**2)/((B**2+1.0)*((G-1.0)*B**2+G+1.0))
      HYPF(B,Z)=Z/(B-Z)
      HYMF(B,Z)=Z/(B+Z)
      GFNF(B)=B/(G*(G-1.0)*(B**2+1.0))

      ROUTINE

      GO TO (20,20,10,10 ),ICODE
   10 HYP1 = 0.0
      HYM2 = 0.0
      ICY = 3
      GO TO 50
   20 IF (Y1) 40, 30, 40
   30 ICY = 2
      GO TO 50
   40 ICY = 1
   50 IFT = 1
      BOLD = 0.0
      ZOLD = 0.0
      SOLD=0.0
      XOLD = 0.0
      YOLD = 0.0
      GO TO (60,70,90), ICY
   60 HYP1=HYPF(B1,Z1)
      GO TO 80
   70 HYP1 = Z2/(B1*Y2)
   80 HYM2=HYMF(B2,Z2)
   90 SHYP1 = HYP1
      SHYM2 = HYM2
      AP1 = APLF(B1,Z1)
      SAP1 = AP1
      AM2 = AMIF(B2,Z2)
      SAM2 = AM2
      E1 = EFNF(Z1)
      SE1 = E1
      E2 = EFNF(Z2)
      SE2 = E2
      F1=FFNF(B1)
      SF1=F1
      F2=FFNF(B2)
      SF2=F2
      GO TO (94,92,94,92),ICODE
   92 G1=0.0
      G2=0.0
```

```
      GO TO 96
   94 G1=GFNF(B1)
      G2=GFNF(B2)
   96 SG1=G1
      SG2=G2
      Z3=(Z1+Z2)*0.5
      S3=0.0
      DO 250 I=1,  MAX
      X3 = (Y2-Y1+AP1*X1-AM2*X2)/(AP1-AM2)
      Y3 = Y2+AM2*(X3-X2)
      GO TO (98,100,98,100),ICODE
   98 S3=S2+((Y3-Y2+Z3*(X2-X3))*(S1-S2))/(Y1-Y2+Z3*(X2-X1))
  100 GO TO (102,102,112,112),ICODE
  102 GO TO(110,104,112) ,ICY
  104 GO TO (106,108),IFT
  106 IFT=2
      FJ3=(X3-X1)*Z2/(B1*Y2)
      GO TO 114
  108 FJ3=(HYP3/Y3+Z2/(B1*Y2))*(X3-X1)*0.5
      GO TO 114
  110 FJ3=HYP1*(X3-X1 )/Y1
      GO TO 114
  112 FJ3=0.0
  114 FJ=E1*Z1-F1*B1+G1*(S3-S1)-FJ3
      FK=E2*Z2+F2*B2-G2*(S3-S2)+HYM2*(X3-X2)/Y2
      Z3=(FJ*F2+FK*F1)/(E1*F2+E2*F1)
      B3=(FK-E2*Z3)/F2
      IF(B3)130,135,130
  130 IF (ABSF((B3-BOLD)/B3)-EPS) 135, 190, 190
  135 IF(Z3)140,145,140
  140 IF (ABSF((Z3-ZOLD)/Z3)-EPS) 145, 190, 190
  145 IF(S3)150,155,150
  150 IF (ABSF((S3-SOLD)/S3)-EPS) 155, 190, 190
  155 IF(X3)160,165,160
  160 IF (ABSF((X3-XOLD)/X3)-EPS) 165, 190, 190
  165 IF(Y3)170,180,170
  170 IF (ABSF((Y3-YOLD)/Y3)-EPS) 180, 190, 190
  180 BN3=B3
      ZN3=Z3
      SN3=S3
      XN3=X3
      YN3=Y3
      RETURN
  190 BOLD = B3
      ZOLD = Z3
      SOLD = S3
      XOLD = X3
      YOLD = Y3
      AP1 = (SAP1+APLF(B3,Z3))*0.5
      AM2 = (SAM2+AMIF(B3,Z3))*0.5
      E3=EFNF(Z3)
      E1=(SE1+E3)*0.5
      E2=(SE2+E3)*0.5
      F3=FFNF(B3)
      F1=(SF1+F3)*0.5
```

```
      F2=(SF2+F3)*0.5
      GO TO (200,210,200,210),ICODE
  200 G3=GFNF(B3)
      G1=(SG1+G3)* 0.5
      G2= (SG2+G3)*0.5
  210 GO TO (220,220,250,250),ICODE
  220 HYP3=HYPF(B3,Z3)
      GO TO (240, 230,250),ICY
  230 HYP1=(HYP3/Y3+SHYP1)*0.5
      GO TO 242
  240 HYP1=(HYP3*(Y1/Y3)+SHYP1 )*0.5
  242 HYM2=((Z3*Y2)/((B3+Z3)*Y3)+SHYM2)*0.5
  250 CONTINUE
      SENSE LIGHT 4
      GO TO 180
      END
```

NOTE: This subroutine differs slightly from the calculation procedure outlined in paragraph 2, page 88 of Ref. [1]. In place of the term

$$\int H^{\pm}(\beta,\zeta)dy/y$$

in Eq. (11) of Ref. [1] we have programmed

$$\int H^{\pm}(\beta,\zeta)dx/y,$$

where $H^{\pm}$ is now defined by

$$H^{\pm}(\beta,\zeta) = \zeta/(\beta \mp \zeta).$$

This eliminates the singularity in $H^{\pm}(\beta,\zeta)$ for Mach lines with zero slope.

```
      SUBROUTINE BOUND (M)

      DIMENSION SPACE(2)
      COMMON SPACE,IL,IU
      K = M-1
      GO TO (100,200), M
  100 GO TO (1000,120,130,140), IU
  120 CALL FIXED(K)
      IF DIVIDE CHECK 121, 125
  121 WRITE OUTPUT TAPE 6, 4
    4 FORMAT(//23H DIVIDE CHECK IN FIXED )
  125 IF(SENSE LIGHT 4)126,128
  126 WRITE OUTPUT TAPE 6,127
  127 FORMAT(52H MAXIMUM NO. OF ITERATIONS EXCEEDED IN FIXED ROUTINE)
  128 CONTINUE
      RETURN
  130 CALL FREE(K)
      IF DIVIDE CHECK 131,132
  131 WRITE OUTPUT TAPE 6,2
    2 FORMAT(//23H DIVIDE CHECK IN FREE  )
  132 IF(SENSE LIGHT 4)133,135
  133 WRITE OUTPUT TAPE 6,134
  134 FORMAT(51H MAXIMUM NO. OF ITERATIONS EXCEEDED IN FREE ROUTINE)
  135 CONTINUE
      RETURN
  140 CALL SHOCK
      IF DIVIDE CHECK 141, 142
  141 WRITE OUTPUT TAPE 6, 5
    5 FORMAT(//23H DIVIDE CHECK IN SHOCK )
  142 IF(SENSE LIGHT 3)143,145
  143 WRITE OUTPUT TAPE 6,144
  144 FORMAT(62H MAX. NO. OF ITERATIONS EXCEEDED IN SHOCK ROUTINE COMPUT
     1ING XI)
  145 IF(SENSE LIGHT 4)146,148
  146 WRITE OUTPUT TAPE 6,147
  147 FORMAT(52H MAXIMUM NO. OF ITERATIONS EXCEEDED IN SHOCK ROUTINE)
  148 CONTINUE
      RETURN
  200 GO TO (210,120,130,2000), IL
  210 CALL AXIS
      IF DIVIDE CHECK 220,230
  220 WRITE OUTPUT TAPE 6,3
    3 FORMAT(//23H DIVIDE CHECK IN AXIS  )
  230 IF(SENSE LIGHT 4)231,233
  231 WRITE OUTPUT TAPE 6,232
  232 FORMAT(51H MAXIMUM NO. OF ITERATIONS EXCEEDED IN AXIS ROUTINE)
  233 CONTINUE
      RETURN
 1000 WRITE OUTPUT TAPE 6, 1,IU
      CALL EXIT
 2000 WRITE OUTPUT TAPE 6, 1,IL
      CALL EXIT
    1 FORMAT(67H PROGRAM STOPPED DURING BOUNDARY DUE TO AN IMPROPER BOUN
     1DARY CODE, I1)
      END
```

```
      SUBROUTINE AXIS
      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, LOW, HIGH, CLIM, EPS, GAMMA, X, Y, BETA, ZETA, S
      APLMIF(A,B,N) = (A*B+(-1.)**N)/(A-B*(-1.)**N)
      E1E2F(A) = 1./(A**2+1.)
      F1F2F(A) = 2.*A**2/((A**2+1.)*((GAMMA-1.)*A**2+GAMMA+1.))
      G1G2F(A) = A/(GAMMA*(GAMMA-1.)*(A**2+1.))
      HPLMIF(A,B,N) = B/(A*B+(-1.)**N)
      IF (CODE-3.) 12, 11, 10
   10 G2 = 0.
      H2 = 0.
      GO TO 25
   11 H2 = 0.
      G2 = G1G2F(BETA(2))
      GO TO 25
   12 IF (CODE-1.) 1000, 20, 15
   15 G2 = 0.
      GO TO 17
   20 G2 = G1G2F(BETA(2))
   17 H2 = HPLMIF(BETA(2),ZETA(2),1)
   25 M = CLIM
      E2=(E1E2F(ZETA(2))+1.)/2.
      F2 = F1F2F(BETA(2))
      DO 50 I=1, M
      BNEW=(E2*ZETA(2)-G2*(S(1)-S(2))-(H2-ZETA(2))/2.)/F2+BETA(2)
      IF (ABSF((BNEW-BETA(1))/BETA(1))-EPS) 100, 100, 30
   30 F2=F1F2F((BNEW+BETA(2))/2.)
      BETA(1) = BNEW
      IF (G2) 35, 50, 35
   35 G2=G1G2F((BNEW+BETA(2))/2.)
   50 CONTINUE
      SENSE LIGHT 4
  100 X(1) = X(2)-2.*Y(2)/(APLMIF(BETA(2),ZETA(2),1)-1./BNEW)
      BETA(1) = BNEW
      RETURN
 1000 WRITE OUTPUT TAPE 6, 1, CODE
    1 FORMAT(49H PROGRAM STOP DUE TO AN IMPROPER FLOW CODE NUMBER,F8.2)
      CALL EXIT
      END
```

```
      SUBROUTINE FIXED (K)

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, LOW, HIGH, A, EPS, GAMMA, X, Y, BETA, ZETA, S
      APLMIF(A,B,M) = (A*B+(-1.)**M)/(A-(-1.)**M*B)
      E1E2F(A) = 1./(A**2+1.)
      F1F2F(A) = 2.*A**2/((A**2+1.)*((GAMMA-1.)*A**2+GAMMA+1.))
      G1G2F(A) = A/(GAMMA*(GAMMA-1.)*(A**2+1.))
      HPLMIF(A,B,M) = B/(A*B+(-1.)**M)
      IF (K) 1000, 10, 20
   10 J1 = N-1
      J3 = N
      GO TO 30
   20 J1 = 2
      J3 = 1
   30 A1 = APLMIF(BETA(J1),ZETA(J1),K)
      E1 = E1E2F(ZETA(J1))
      F1 = F1F2F(BETA(J1))
      IF (CODE-2.) 80, 40, 50
   40 G1 = 0.
      GO TO 85
   50 IF (CODE-3.) 1000, 70, 60
   60 G1 = 0.
   65 H1 = 0.
      GO TO 90
   70 G1 = G1G2F(BETA(J1))
      GO TO 65
   80 G1 = G1G2F(BETA(J1))
   85 H1 = HPLMIF(BETA(J1),ZETA(J1),K)
   90 X(J3) = 1.E10
      Y(J3) = 1.E10
      BETA(J3) = 1.E10
      ZETA(J3) = 1.E10
      XNEW=X(J1)
      LIM = A
      DO 200 I=1, LIM
      XNEW=(Y(J1)-WALL(XNEW,K)+XNEW*WALPR(XNEW,K)-X(J1)*A1)/(WALPR(XNEW,
     1K) - A1)
      YNEW = WALL(XNEW,K)
      ZNEW=WALPR(XNEW,K)
      BNEW = ((-1.)**K*E1*(ZNEW-ZETA(J1))-G1*(S(J3)-S(J1))+H1*(YNEW-
     1Y(J1))/Y(J1))/F1+BETA(J1)
      IF (ABSF((X(J3)-XNEW)/XNEW)-EPS) 100, 100, 150
  100 IF (ABSF((Y(J3)-YNEW)/YNEW)-EPS) 110, 110, 150
  110 IF (ABSF((ZETA(J3)-ZNEW)/ZNEW)-EPS) 120, 120, 150
  120 IF (ABSF((BETA(J3)-BNEW)/BNEW)-EPS) 210, 210, 150
  150 X(J3) = XNEW
      Y(J3) = YNEW
      BETA(J3) = BNEW
      ZETA(J3) = ZNEW
      B1=(BNEW + BETA(J1))/2.
      Z1=(ZNEW + ZETA(J1))/2.
      A1=APLMIF(B1,Z1,K)
      F1=F1F2F(B1)
```

```
      E1=E1E2F(Z1)
      IF (G1) 160, 170, 160
  160 G1=G1G2F(B1)
  170 IF (H1) 180, 200, 180
  180 H1=(HPLMIF(B1,Z1,K))*(2.*Y(J1))/(Y(J1)+YNEW)
  200 CONTINUE
      SENSE LIGHT 4
  210 RETURN
 1000 WRITE OUTPUT TAPE 6, 1, CODE
      CALL EXIT
    1 FORMAT(42H1ERROR EXIT FROM FIXED SUBROUTINE. CODE = F7.4)
      END
```

```
      FUNCTION WALL (X,K)
      IF (K) 1000, 10, 20
   10 WALL = 1. + X*X/5.
      RETURN
   20 WALL= .1 + .05004166*X - .01001042*X*X
      RETURN
 1000 WRITE OUTPUT TAPE 6, 1, K
      CALL EXIT
    1 FORMAT(38H1ERROR EXIT FROM WALL SUBROUTINE. K = ,I6)
      END
```

```
      FUNCTION WALPR(X,K)
      IF (K) 1000,10,20
   10 WALPR = .4*X
      RETURN
   20 WALPR = .05004166 - .02002084*X
      RETURN
 1000 WRITE OUTPUT TAPE 6,1,K
      CALL EXIT
    1 FORMAT(38H1ERROR EXIT FROM WALL SUBROUTINE. K = ,I6)
      END
```

```
      SUBROUTINE SHOCK

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, XYZ, ZYX, C, EPS, GAMMA, X, Y, BETA, ZETA, S, BO,
     1XI
      APLMIF(A,B) = (A*B+1.)/(A-B)
      E1E2F(A) = 1./(A*A+1.)
      F1F2F(A) = 2.*A*A/((A*A+1.)*((GAMMA-1.)*A*A+(GAMMA+1.)))
      G1G2F(A) = A/(GAMMA*(GAMMA-1.)*(A*A+1.))
      HPLMIF(A,B) = B/(A*B+1.)
      C1 = (GAMMA-1.)*B0*B0+GAMMA+1.
      C2 = C1+2.*(B0*B0+1.)
      GA1=(GAMMA+1.)/(GAMMA-1.)
      C3= C1/(GAMMA - 1.) + GA1*B0*B0
      X3 = 1.E+10
      Y3 = 1.E+10
      Z3 = 1.E+10
      B3 = 1.E+10
      S3 = 1.E+10
      SB=BETA(N-1)
      SZ=ZETA(N-1)
      SX=X(N-1)
      SY=Y(N-1)
      SXI=XI
      A1 = APLMIF(BETA(N-1),ZETA(N-1))
      D1= (Y(N-1) - Y(N))/(X(N-1) - X(N))
      XC= X(N)+2.0*(A1-XI)*(X(N-1)-X(N)) /(A1-D1)
      YC=Y(N-1)+D1*(XC-X(N-1))
      D2=(XC-X(N))/(X(N-1)-X(N))
      BETA(N-1)=BETA(N-1)*D2+BETA(N)*(1.-D2)
      ZETA(N-1)=ZETA(N-1)*D2+ZETA(N)*(1.-D2)
      S(N-1)=S(N-1)*D2+S(N)*(1.-D2)
      X(N-1)=XC
      Y(N-1)=YC
      E1 = E1E2F(ZETA(N-1))
      F1 = F1F2F(BETA(N-1))
      A1 = APLMIF(BETA(N-1),ZETA(N-1))
      G1 = G1G2F(BETA(N-1))
      IF (CODE-3.)55,35,35
   35 H1 = 0.
      GO TO 60
   55 H1 = HPLMIF(BETA(N-1),ZETA(N-1))
   60 L=C
      DO 300 I=1, L
      XNEW = (Y(N-1)-Y(N)+XI*X(N)-A1*X(N-1))/(XI-A1)
      YNEW = Y(N-1)+A1*(XNEW-X(N-1))
      XI2 = XI
      DO 70 J=1, L
      XIS = XI2*XI2
      FD1 = C1*XIS + 2.
      FD2 = XIS + 1.
      FD3 = C3*XIS - 1.
      FD4 = FD1 + C2 - 2.
      SF1 = 2./(XIS*FD1)
```

```
      SF2 = (C3 + 1.)/(FD2*FD3)
      ZNEW = 2.*(BO*BO*XIS - 1.)/(XI2*FD4)
      SNEW = GAMMA*LOGF(FD1/((GAMMA + 1.)*XIS*(BO*BO + 1.)))
     1  +LOGF(FD3/(GA1*FD2))
      BNSQ = GA1*(-1. + (C1*FD2*GA1*XIS*(BO*BO +1.))/(FD1*FD3))
      BNEW = SQRTF(BNSQ)
      ZP=2.*(2.*BO*BO-C1*XI2*ZNEW)/FD4-ZNEW/XI2
      SP = 2.*XI2*(SF2 - GAMMA*SF1)
      BP = XI2*(BNSQ + GA1)*(SF1 - SF2)/BNEW
      FXI = E1*(ZNEW - ZETA(N-1)) - F1*(BNEW - BETA(N-1))
     1   -G1*(SNEW - S(N-1)) + H1*2.*(YNEW - Y(N-1))/(YNEW+Y(N-1))
      FXIP = E1*ZP - F1*BP - G1*SP
      XI3 = XI2 - FXI/FXIP
      IF (ABSF((XI3-XI2)/XI3)-EPS) 100,100,65
   65 XI2 = XI3
   70 CONTINUE
      SENSE LIGHT 3
  100 IF (ABSF((X3-XNEW)/XNEW)-EPS) 110, 110, 200
  110 IF (ABSF((Y3-YNEW)/YNEW)-EPS) 120, 120, 200
  120 IF (ABSF((B3-BNEW)/BNEW)-EPS) 130, 130, 200
  130 IF (ABSF((Z3-ZNEW)/BNEW)-EPS) 140, 140, 200
  140 IF(SNEW)145,150,145
  145 IF(ABSF((S3-SNEW)/SNEW)-EPS)150,150,200
  150 X(N) = XNEW
      Y(N) = YNEW
      BETA(N) = BNEW
      ZETA(N) = ZNEW
      S(N) = SNEW
      XI = XI3
      X(N-1)=SX
      Y(N-1)=SY
      ZETA(N-1)=SZ
      BETA(N-1)=SB
      RETURN
  200 X3 = XNEW
      Y3 = YNEW
      Z3 = ZNEW
      B3 = BNEW
      S3 = SNEW
      XI=(SXI+XI3)/2.
      B1=(BNEW+BETA(N-1))/2.
      Z1=(ZNEW+ZETA(N-1))/2.
      A1=APLMIF(B1,Z1)
      E1=E1E2F(Z1)
      F1=F1F2F(B1)
      G1=G1G2F(B1)
      IF (H1) 215, 300, 215
  215 H1= HPLMIF(B1,Z1)
  300 CONTINUE
      SENSE LIGHT 4
      GO TO 150
      END
```

```
      SUBROUTINE FREE (M)

      DIMENSION X(1000), Y(1000), BETA(1000), ZETA(1000), S(1000)
      COMMON N, CODE, LOW, HIGH, A, EPS, GAMMA, X, Y, BETA, ZETA, S
      APLMIF(A,B,K) = (A*B+(-1.)**K)/(A-B*(-1.)**K)
      E1E2F(A) = 1./(A**2+1.)
      F1F2F(A) = 2.*A*A/((A*A+1.)*((GAMMA-1.)*A*A+(GAMMA+1.)))
      G1G2F(A) = A/(GAMMA*(GAMMA-1.)*(A*A+1.))
      HPLMIF(A,B,K) = B/(A*B+(-1.)**K)
      IF (M) 1000, 10, 20
   10 J1 = N-1
      J2 = N
      K = 0
      GO TO 30
   20 J1 = 2
      J2 = 1
      K = 1
   30 B1=(BETA(J1)+BETA(J2))/2.
      A1=APLMIF(B1,ZETA(J1),K)
      E1 = E1E2F(ZETA(J1))
      F1=F1F2F(B1)
      IF (CODE-2.) 80, 40, 50
   40 G1 = 0.
      GO TO 90
   50 IF (CODE-3.) 1000, 60, 70
   60 H1 = 0.
      G1 = G1G2F(B1)
      GO TO 100
   70 G1 = 0.
      H1 = 0.
      GO TO 100
   80 G1 = G1G2F(B1)
   90 H1 = HPLMIF(B1,ZETA(J1),K)
  100 L = A
      ZETAC= ZETA(J2)
      X3 = 1.E10
      Y3 = 1.E10
      Z3 = 1.E10
      DO 300 I=1, L
      XNEW = (Y(J1)-Y(J2)+X(J2)*ZETAC   -X(J1)*A1)/(ZETAC   -A1)
      YNEW = Y(J1)+A1*(XNEW-X(J1))
      ZNEW = ZETA(J1)+((-1.)**K*F1*(BETA(J2)-BETA(J1))+(-1.)**K*G1*
     1(S(J2)-S(J1))-(-1.)**K*H1*(YNEW-Y(J1))/Y(J1))/E1
      IF (ABSF((ZNEW-Z3)/ZNEW)-EPS) 150, 150, 210
  150 IF (ABSF((XNEW-X3)/XNEW)-EPS) 200, 200, 210
  200 IF (ABSF((YNEW-Y3)/YNEW)-EPS) 500, 500, 210
  210 X3 = XNEW
      Y3 = YNEW
      Z3 = ZNEW
      Z1=(ZETA(J1)+ZNEW)/2.
      A1=APLMIF(B1,Z1,K)
      E1=E1E2F(Z1)
      IF (H1) 260, 250, 260
  250 H1 = 0.
```

```
      GO TO 270
 260  H1=(HPLMIF(B1,Z1,K))*Y(J1)*2./(Y(J1)+YNEW)
 270  ZETAC=(ZETA(J2)+ZNEW)/2.
 300  CONTINUE
      SENSE LIGHT 4
 500  ZETA(J2) = ZNEW
      X(J2) = XNEW
      Y(J2) = YNEW
      RETURN
1000  WRITE OUTPUT TAPE 6, 1, M
      CALL EXIT
    1 FORMAT(38H1ERROR EXIT FROM FREE SUBROUTINE. M = I6)
      END
```

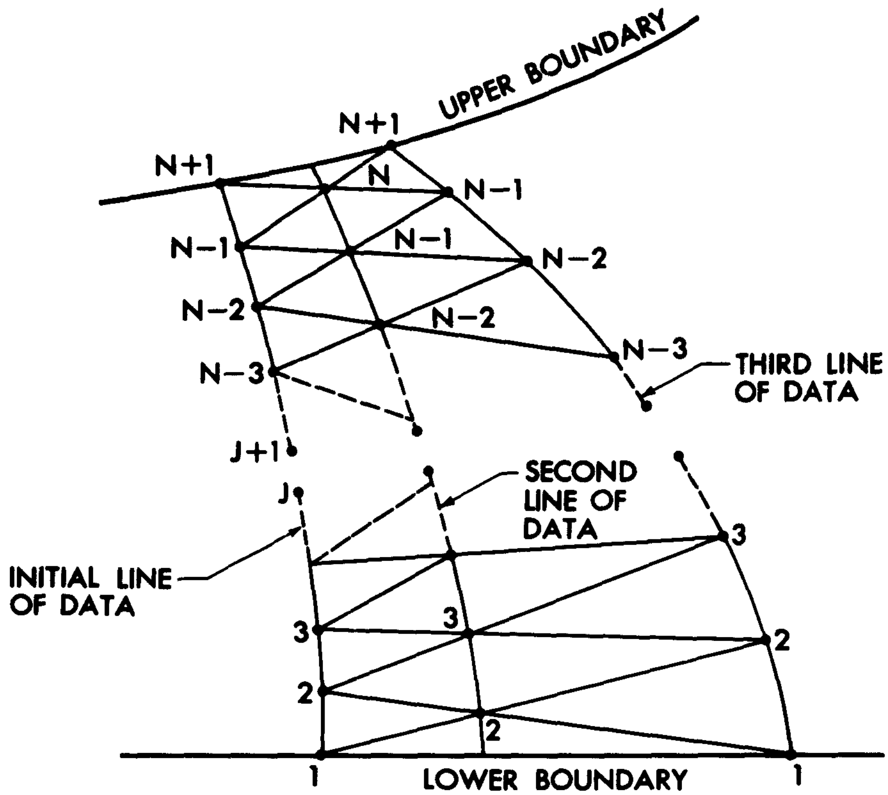

Fig. 1 Illustration for the method of computing and storing points of the Mach net.

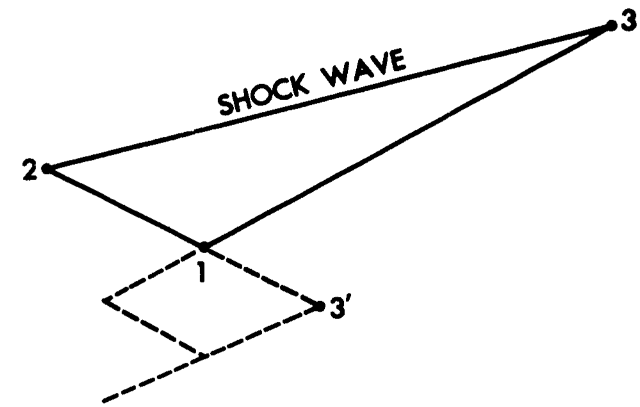

Fig. 2 Illustration for the method of computing new point along a shock.

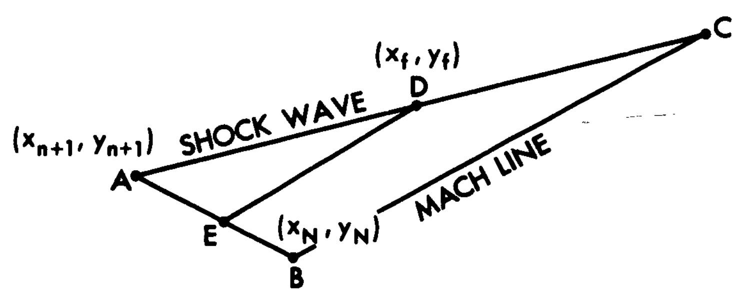

Fig. 3 Illustration for method of reducing net size in the region near a shock.

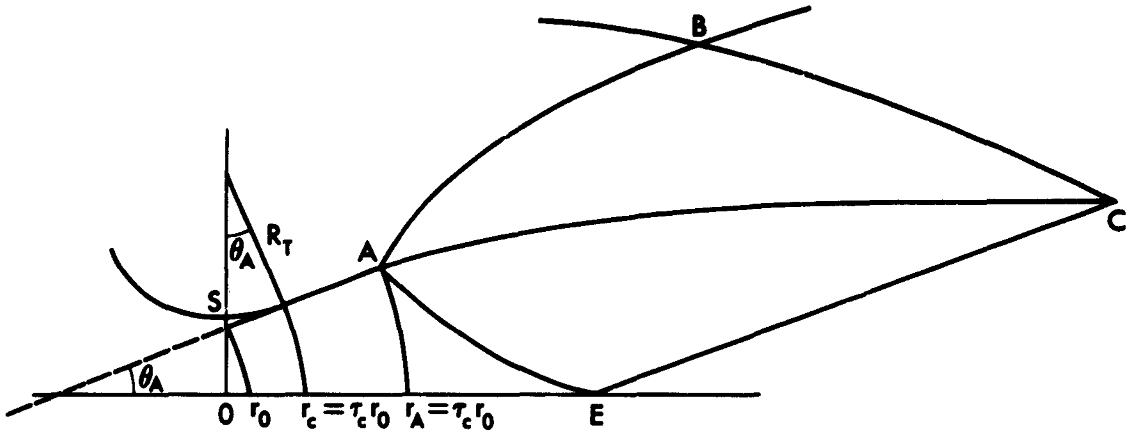

Fig. 4 Illustration for method of computing axially symmetric perfect nozzles using conical source flow (Ref.[2])

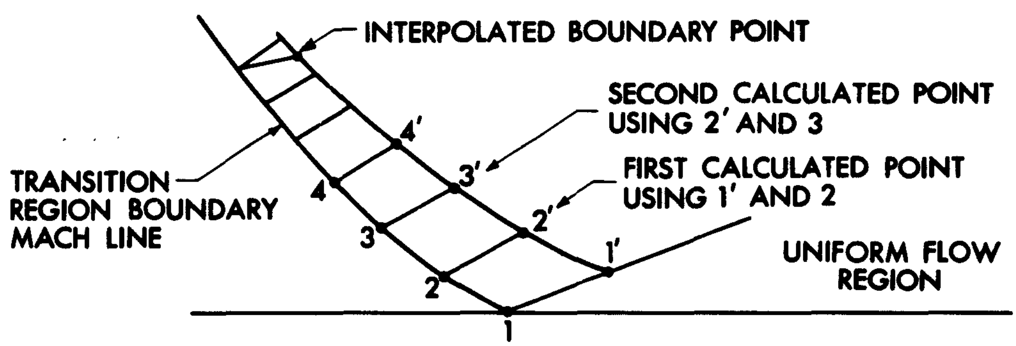

Fig. 5 Illustration for strip method of computing Mach net of the transition region of an axially symmetric nozzle.

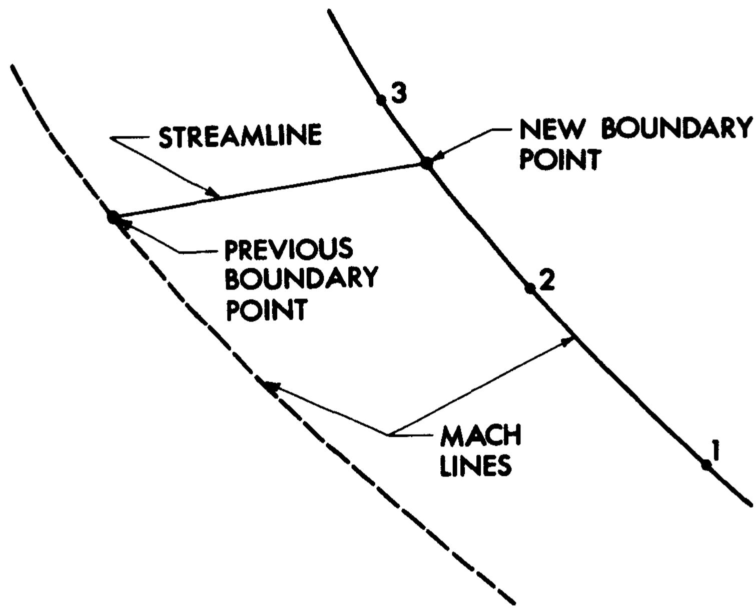

Fig. 6 Illustration for the computation of the wall streamline through the transition region of an axially symmetric nozzle.

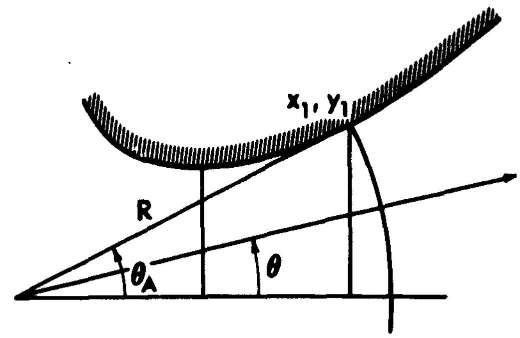

Fig. 7 Initial data line near throat of a nozzle using source flow.

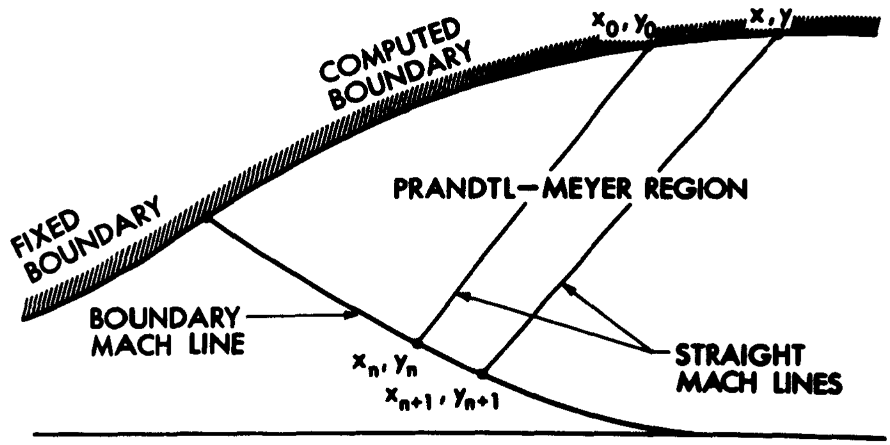

Fig. 8 Illustration for the calculation of the wall streamline through the Prandtl-Meyer region of a two dimensional nozzle.